Fast and Accurate Molecular Dynamics Simulation of a Protein Using a Special-Purpose Computer

YUTO KOMEIJI,[1] MASAMI UEBAYASI[2], RYO TAKATA,[3] AKIHIRO SHIMIZU,[3] KEIJI ITSUKASHI,[3] MAKOTO TAIJI[4]

[1]*Electrotechnical Laboratory, 1-1-4 Umezono, Tsukuba-shi, Ibaraki, 305, Japan*
[2]*National Institute of Bioscience and Human-Technology, 1-1-3 Higashi, Tsukuba-shi, Ibaraki 305, Japan*
[3]*ITL Corp., 3-36-19 Tamagawa, Chofu-shi, Tokyo 182, Japan*
[4]*College of Arts and Science, The University of Tokyo, Komaba, Meguro-ku, Tokyo, 153, Japan*

Received 7 October 1996; accepted 24 March 1997

ABSTRACT: The rapid and accurate molecular dynamics simulation of biomolecules was made possible by a special purpose computer, MD-GRAPE (GRAvity PipE for Molecular Dynamics), which computes arbitrary central force and potential. A program package for the molecular dynamics simulation of biological macromolecules was developed (PEACH Program for Energetic Analysis of bioCHemical molecules), which used MD-GRAPE for computation of the nonbonded interactions (van der Waals, and direct or Ewald summation of the electrostatic) without a cutoff scheme. A multiple time step integrator from the literture was implemented in PEACH to save computation time. Nanosecond order molecular dynamics simulations of a fully solvated histidine-containing phosphocarrier protein ($\sim$10,000 atoms) were performed in a spherical (direct summation of the electrostatic) or a periodic (Ewald summation) boundary with or without Nose–Hoover isothermal algorithm. The trajectories thus obtained without the nonbonded cutoff were quite stable, indicating the usefulness of the PEACH-GRAPE system constructed in this study. © 1997 John Wiley & Sons, Inc. *J Comput Chem* **18**: 1546–1563, 1997

Keywords: molecular dynamics; special-purpose computer; multiple time step method; electrostatic interaction; HPr

Correspondence to: Y. Komeiji; e-mail: komeiji@etl.go.jp

CCC 0192-8651 / 97 / 121546-18

Introduction

Molecular dynamics (MD) simulation is a powerful tool for examining the structural, dynamical, and functional problems of biomolecules such as proteins.[1] The size and simulated time of the molecules are limited, however, because biomolecular systems often consist of so many atoms that their simulation requires enormous computer resources.

The most time consuming part of the simulation is the electrostatic interactions. Therefore, various cutoff methods have long been in use; for instance, only the interactions between atom pairs within a specified cutoff radius (usually 8–15 Å) are considered. Although such methods result in great computational savings for MD of large biomolecules, serious problems have been reported in the recent literature, for instance, the abrupt cutoff results in different temperatures of the protein and the solvent.[2–5] Charged residues of a protein undergo unacceptably large fluctuations.[6] The stability of a solvated α-helix is highly dependent on the value of the cutoff radius.[7] Use of the cutoff method is also the main cause of the computational error in free energy calculations.[8,9] These artifacts were not unexpected, considering the crucial role played by the electrostatic interaction in the biomolecular systems. Hence, avoidance of a cutoff should increase the reliability of the computational results.

In an N-body simulation, the computational cost of the electrostatic interaction increases in proportion to $O(N^2)$. There are two ways to avoid the cutoff scheme without increasing the computation time. One is the development of algorithms that reduce the computational demand to $O(N \log N)$ or $O(N)$.[10–17] The other is use of a special purpose computer for tasks proportional to $O(N^2)$.[5,18–22] Both have their merits and shortcomings, but we took the latter approach in this study for the future development of a massive parallel MD computer. Also, it is possible to use a special purpose computer with some of the $O(N \log N)$ algorithms.[23]

The machine we used, MD-GRAPE,[24] is one of the series of special-purpose computers named GRAPE (GRAvity PipE).[25,26] As its name indicates, GRAPE was originally developed for gravitational problems, but its application has been extended to other N-body problems including MD. GRAPE is usually connected to a host (a personal computer or a workstation) through a bus adapter (Fig. 1). GRAPE performs tasks proportional to $O(N^2)$ such as the electrostatic or van der Waals (vdW) interactions, while the host performs those proportional to $O(N)$, such as bonded interactions and time integration. The data are transmitted between the host and GRAPE via the bus adapter. See ref. 20 for comparison of GRAPE with other types of special-purpose computers. The first GRAPE for MD was GRAPE2A,[20] which has a peak speed of 180 Mflops. A large scale integration (LSI) named MD-chip was developed based on the architecture of GRAPE2A.[24] The MD-chip computes forces acting on six atoms simultaneously by a virtual multipipeline method at a peak speed of 0.75 Gflops. The MD-chip handles arbitrary central force and potential, as well as the Ewald force and potential. The precision is between single (32 bit) and double (64 bit). MD-GRAPE is a parallel computer carrying several MD-chips.[24,27]

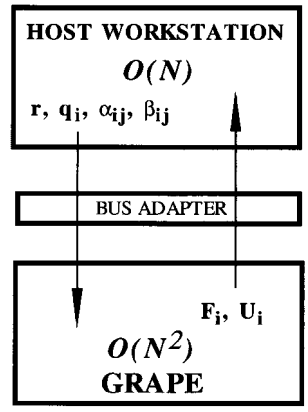

FIGURE 1. The hardware architecture of the MD-GRAPE system. The host performs tasks proportional to $O(N)$, while GRAPE performs those proportional to $O(N^2)$. The data are transmitted via the bus adapter.

We previously investigated the performance and precision of MD-GRAPE as applied to MD.[27] The MD-GRAPE containing eight MD-chips had an effective speed of 6 Gflops, as expected from the peak speed of MD-chips ($0.75 \times 8 = 6$ Gflops).

We also developed a software package, PEACH (Program for Energetic Analysis of bioCHemical molecules), for MD of biomolecules with MD-GRAPE and gave a preliminary description of the program.[27] The precision of the trajectory generated by the PEACH-GRAPE system was comparable to that by a full double precision computation. The computation speed was 0.8–6 (s) computation time/step for MD of several solvated proteins ($N \sim 3000$–$27{,}000$) without using the cutoff

scheme. The same calculations took 20–200 times longer on the host workstation (DEC Alphastation 3000/700/225).

In this article we present a more detailed account of the PEACH-GRAPE system. The Ewald[28] mode is described for MD in a periodic boundary condition. The implementation of the multiple time step method[29] and the Nose–Hoover thermostat[30] in PEACH is reported. Finally, results of nanosecond order MD simulations of a fully solvated protein, HPr (histidine containing phosphocarrier protein),[31] are given to demonstrate the performance of the PEACH-GRAPE system. HPr is a small (88 amino acid residues, $M_r \sim 80{,}000$) heat-stable protein, and hence we considered it suitable for testing the stability and reliability of a new MD system. The convergence and equilibrium properties of the trajectories are investigated.

Special-Purpose Computer MD-GRAPE

HARDWARE ARCHITECTURE OF MD-GRAPE

The hardware system of MD-GRAPE was already described,[27] and we give only an outline (Fig. 1). The host workstation was either a DEC Alphastation model 3000/700/225 (164 Mflops at peak) or model 3000/600/5 (343 Mflops at peak). Data listed in Tables I and II were taken by using the latter. MD-GRAPE (or ITL-MD one, ITL Corp., Tokyo) had eight MD-chips and was connected to the host through a bus adapter (DEC Turbo-VME, Model 497, BIT3). Communication between the host and MD-GRAPE was governed by an embedded controller (MVME162, Motorola) on MD-GRAPE.

INTERACTIONS COMPUTED BY MD-GRAPE

The MD-chip computes arbitrary central force and potential having the forms below:

$$\mathbf{f}_i = \sum_j \alpha_{ij} \Phi\left(\beta_{ij} r_{ij}^2\right) \mathbf{r_{ij}}, \tag{1}$$

$$u_i = \sum_j \gamma_{ij} \Psi\left(\delta_{ij} r_{ij}^2\right), \tag{2}$$

where $\mathbf{r_{ij}} = \mathbf{r_i} - \mathbf{r_j}$. The functions $\Phi(x)$ and $\mathrm{X}(x)$, and the coefficients $a_{ij} \sim \delta_{ij}$ differ depending on the interactions to compute. MD-GRAPE currently handles the electrostatic, Lennard–Jones (referred to as vdW throughout this article), and H-bond interactions, whose forces and potentials are listed below, as well as the Ewald real-space (r-space) summation described later.

$$\text{Electrostatic:} \quad \frac{u_i}{q_i} = \sum_j \frac{q_j}{r_{ij}}, \quad \frac{\mathbf{f_i}}{q_i} = \sum_j \frac{q_j}{r_{ij}^3} \mathbf{r_{ij}} \tag{3}$$

where q_i is the partial charge of atom i.

$$\text{vdw:} \quad u_i = \sum_j \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^6} \right),$$

$$\mathbf{f}_i = \sum_j \left(\frac{12\,A_{ij}}{r_{ij}^{14}} - \frac{6B_{ij}}{r_{ij}^8} \right) \mathbf{r_{ij}}. \tag{4}$$

TABLE I.
Determination of Time Steps for drRESPA.[a]

	Δt(fs)			Sphere		Box	
Condition	H[b]	M[c]	S[d]	$\mathrm{RMSF_r}$ (E_{tot})	Time[a] (s / fs)	$\mathrm{RMSF_r}$ (E_{tot})	Time (s / fs)
1	0.5	0.5	0.5	4.0×10^{-4}	2.9	1.5×10^{-5}	5.3
2	0.5	1.0	1.0	6.5×10^{-4}	1.6	2.0×10^{-5}	2.8
3	0.5	1.0	2.0	4.7×10^{-4}	1.1	8.2×10^{-5}	1.7
4[f]	0.5	2.0	2.0	3.9×10^{-4}	0.9	8.3×10^{-5}	1.5
5	0.5	2.0	4.0	1.4×10^{-3}	0.6	1.3×10^{-3}	0.9

[a]The 10-ps configuration of each trajectory was used as the initial configuration. The system was simulated for 5 ps using a microcanonical ensemble with various time steps. Then, the $\mathrm{RMSF_r}(E_{tot})$ and the computation time was examined.
[b]Hard forces (H): bond, angle, and torsion.
[c]Medium force (M): vdW.
[d]Soft force (S): electrostatic.
[e]Computation time (s) consumed per 1 fs MD.
[f]This condtion was adopted for further MD simulations.

TABLE II.
Details of Computation Time (s) Consumed Per 1 fs MD.

	Sphere (Direct Sum)		Box (Ewald Sum)	
Interaction	Host[a]	GRAPE[b]	Host	GRAPE
Bond	0.024		0.025	
Angle	0.043		0.045	
Torsion	0.022		0.023	
1 −4 nonbonded	0.011		0.011	
VDW	0.022	0.201	0.023	0.215
Electrostatic	0.023	0.453		
r-space			0.024	0.485
k-space			0.009	0.540
Constraint[c]	0.005			
Miscellaneous	0.051		0.076	
Subtotal	0.20	0.65	0.21	1.24
Total	0.86		1.45	
8-Å cutoff MD[d] PEACH	1.84		2.15	
AMBER	0.96		0.58	

[a]Time consumed by the host (DEC Alphastation model 3000 / 600 / 5).
[b]Time consumed by MD-GRAPE and the interface library.
[c]Force imposed to constrain the solvent within the sphere.
[d]MD with residue-based cutoff method was performed by the general-computer version of PEACH or by AMBER 4.1[45] on the same workstation for comparison of the speed with the PEACH-GRAPE system. The cut-off radius was 8 Å and the nonbonded-list was updated every 20 fs. The SANDER module of AMBER 4.1 was compiled by DEC FORTRAN 77 under the compiler options specified in AMBER, and SHAKE[46] was used instead of RESPA to allow a time step of 2 fs.

$$\text{H bond:}\quad u_i = \sum_j \left(\frac{C_{ij}}{r_{ij}^{12}} - \frac{D_{ij}}{r_{ij}^{10}} \right),$$

$$\mathbf{f_i} = \sum_j \left(\frac{12C_{ij}}{r_{ij}^{14}} - \frac{10D_{ij}}{r_{ij}^{12}} \right)\mathbf{r_{ij}} \quad (5)$$

where A_{ij}–D_{ij} are force field parameters. The parameters are transformed into a_{ij}–δ_{ij} for computation by GRAPE as listed in Table III.

MD-chip also handles the following equation for computation of Ewald wave number space (*k*-space) force and potential, which will be described later.

$$\mathbf{w_i} = \sum_j \varepsilon_{ij}\Theta(\lambda_{ij}\mathbf{k_i} \cdot \mathbf{r_{ij}}). \quad (6)$$

DIRECT SUMMATION OF ELECTROSTATIC INTERACTION

We developed a FORTRAN interface library for accessing MD-GRAPE. Below we give an excerpt of FORTRAN-like pseudocode for direct summation of the electrostatic force to illustrate how to use MD-GRAPE. Comments are preceded by an exclamation point (!). The names of the interface subroutines are prefixed by M1.

```
Program for GRAPE (1): Electrostatic force (direct summation)
integer:: n          ! number of the atoms
real:: r(3, n)       ! coordinates of the atoms
                     ! (r(1, i),  r(2, i),  r(3, i)) =
                     ! (x, y, z) of atom (i)
real:: f(3, n)       ! force acting on the atoms
real:: q(n)          ! partial charges of the atoms
logical:: ifpbc      ! flag set true if periodic
                     ! boundary condition
real:: rmax, fmax, box
                     ! see the text for description
integer:: iovf       ! flag to indicate if overflow
                     ! occurred

call M1presetmode(1)  ! set electrostatic force
                     ! mode
if (ifpbc) then
  call M1setcellsize(box/2, fmax)  ! see text
                     ! for description
else
  call M1setautoscale(rmax, imax)  ! see text
                     ! for description
```

TABLE III.
Interactions Computed by MD-GRAPE.

Interaction	F / U	Mode	$\Phi(x) / \Psi(x)$	$\alpha_{ij} / \gamma_{ij}$	β_{ij} / δ_{ij}
Coulomb	F	1	$x^{-3/2}$	q_j	1
	U	2	$x^{-1/2}$	q_j	1
VDW	F	3	$x^{-7} - x^{-4}$	$(B_{ij} / 2A_{ij})^{1/3}$	$(2B_{ij}^6 / A_{ij}^4)^{1/3}$
	U	4	$x^{-6} - x^{-3}$	B_{ij}^2 / A_{ij}	$(B_{ij} / A_{ij})^{1/3}$
H bond	F	5	$x^{-7} - x^{-6}$	$5D_{ij} / 6C_{ij}$	$(10D_{ij})^7 / (12C_{ij})^6$
	U	6	$x^{-6} - x^{-5}$	D_{ij}^6 / C_{ij}^5	D_{ij} / C_{ij}
Ewald (real)	F	7	$x^{-3/2}\mathrm{erfc}(x^{1/2}) + 2\pi^{-1/2}x^{-1}e^{-x}$	q_j	$1 / \eta^2$
	U	8	$x^{-1/2}\mathrm{erfc}(x^{1/2})$	q_j	$1 / \eta^2$
(wave)		9	See text		

[a]F stands for force, and U stands for potential.

```
end if
call M1setxjn(1, n, r(1, 1))  ! send coordinates of
                              ! atoms 1−n
                              ! to GRAPE
call M1setqxn(1, n, q(1))     ! send charges of
                              ! atoms 1−n
                              ! to GRAPE
call M1execn(r(1, 1), n, n, 0, f(1, 1), iovf)! calcu-
                              ! late force
do i = 1, n
  f(:, i) = q(i)∗ f(:, i)     ! multiply force by
                              ! q(i)
end do
```

M1presetmode(1) sets GRAPE to the electrostatic force mode (Table III). The input and output of MD-GRAPE are fixed-point formats,[24] and it is necessary to set the decimal point before computation either by M1setautoscale for nonperiodic or by M1setcellsize for periodic boundary condition. The argument $f$max is the presumable maximum of $|f(m, i)|$ $(m = 1-3,\ i = 1-n)$, and $r$max of M1setautoscale is maximum of $\{|r(m, i)|, |r(m, i) - r(m, j)|\}$ $(m = 1-3,\ i, j = 1-n)$. Setting half of the box size (box/2) by M1setcellsize introduces the periodic boundary condition; namely, a minimal image is automatically selected.[24] M1setxjn and M1setqxn send the coordinates and charges of atoms $\sim n$ to the memory of MD-GRAPE. M1execn calculates either force or potential, depending on the mode (Table III). MD-GRAPE avoids self-interaction automatically. The arguments of M1execn are demonstrated as follows.

```
subroutine M1execn(r, n, I, type, force, iovf)
! Input:
  real:: r(3, n)      ! coordinates of the atoms
                      ! forces acting on
                      ! which are computed
  integer:: n         ! number of the above atoms
  integer:: m         ! number of the interacting
                      ! atoms
  integer:: type      ! atom type index of the
                      ! atoms to compute
                      ! (dummy in the Coulomb
                      ! mode)
! Output
  real:: f(3, n)      ! computed force or potential
  integer:: iovf      ! flag to indicate overflow of
                      ! force
                      ! 1 is returned if overflow
                      ! occurred,
                      ! and 0 if not.
```

VDW INTERACTION

The parameters a_{ij}–δ_{ij} [eqs. (1) and (2)] depend only on j in the electrostatic interaction, but they depend both on i and j in vdW or H-bond interactions (Table III). For such interactions, the program becomes a little more complicated because the atoms handled by MD-GRAPE simultaneously should belong to the same atom type.

```
Program for GRAPE (2): vdW force
integer:: ntype      ! number of atom types
real:: A(ntype, ntype), B(ntype, ntype)
```

```
                              ! force field parameters
integer::  type(n)            ! atom type index of the
                              ! atoms
integer::  ntype              ! number of atom types
integer::  point(ntype)
                              ! pointer to the first atom
                              ! of each atom type
(variables common with program 1 are omitted)

call M1presetmode(3)          ! set vdW force
                              ! mode
if (ifpbc) then
  call M1setcellsize(box/2, fmax)
else
  call M1setautoscale(rmax, fmax)
end if
do itype = 1, ntype
  call M1setgrascalen(i, ntype, A(1, itype), B(1,
                                itype))
      ! send force parameters of each atom type
end do
call M1setxjn(1, n, r(1,1))   ! send coordinates
                              ! of atoms 1-n
                              ! to GRAPE
call M1setcjn(1, n, type(1))
                              ! send atom type
                              ! index of atoms
                              ! 1-n
                              ! to GRAPE
do k = 1, ntype
  i1 = point(k)               ! the first atom
                              ! having type (k)
  i2 = point(k + 1) - 1       ! the last atom
                              ! having type (k)
  m = i2 - i1 + 1             ! the number of
                              ! atoms
                              ! belonging to type
                              ! (k)
  call M1execen(r(1, i1), m, n, type(k), f(1, 1),
                                iovf)
end do
```

Before using program 2, the coordinates of the atoms must be sorted according to their atom types. The force field parameters are sent to GRAPE by M1setgrascalen. Then atom type index of each atom is sent to GRAPE by M1setcjn. This time the argument "type" is not a dummy. M1execn computes force by choosing appropriate A(i, j) and B(i, j) for i and j.

EWALD SUMMATION OF ELECTROSTATIC

The Ewald method is a conventional but powerful way to treat electrostatic interaction precisely in a periodic boundary.[28] Let us consider a cubic cell of length L containing several atoms. Total electrostatic energy (U_{tot}) is defined as follows:

$$U_{\mathrm{tot}} = \frac{1}{2}\sum_{\mathbf{n}}\sum_{i} q_i \sum_{j}{}' \frac{q_j}{|\mathbf{r_i} - \mathbf{r_j} - L\mathbf{n}|}, \qquad (7)$$

where $\mathbf{n} = (n_x, n_y, n_z)$ is a vector whose components are integers ($n_x, n_y, n_z = 0, \pm 1, \pm 2, \ldots$). Σ' indicates that the term $i = j$ should be omitted when $\mathbf{n} = (0, 0, 0)$. The summation (7) can be decomposed into two summations and a constant.

$$U_{\mathrm{real}} = \frac{1}{2}\sum_{\mathbf{n}}\sum_{i} q_i \sum_{j}{}' \frac{q_j}{|\mathbf{r_i} - \mathbf{r_j} - L\mathbf{n}|} \times \mathrm{erfc}\left(\frac{|\mathbf{r_i} - \mathbf{r_j} - L\mathbf{n}|}{\eta}\right)$$

real space (r-space) (8)

$$U_{\mathrm{wave}} = \frac{1}{2}\sum_{\mathbf{k}} \frac{\exp\left(-\frac{\pi^2\eta^2\mathbf{k}^2}{L^2}\right)}{\frac{\mathbf{k}^2}{L^2}} \times \left\{\left(\sum_i q_i \cos\frac{2\pi\mathbf{k}\cdot\mathbf{r_i}}{L}\right)^2 + \left(\sum_i q_i \sin\frac{2\pi\mathbf{k}\cdot\mathbf{r_i}}{L}\right)^2\right\}$$

wave number space (k-space) (9)

$$U_{\mathrm{self}} = -\frac{1}{\pi^{1/2}\eta}\sum_i q_i^2 \ \text{self-term} \qquad (10)$$

The vector $\mathbf{k}$ is similar to $\mathbf{n}$, and the components are integers. The parameter h determines the balance among the three terms. The above potentials satisfy $U_{\mathrm{tot}} = U_{\mathrm{real}} + U_{\mathrm{wave}} + U_{\mathrm{self}}$.

MD-GRAPE treats the Ewald summation similarly to ref. 32. U_{self} is computed by the host at the beginning of the simulation. U_{real} and the derived force are computed by GRAPE in nearly the same way as the direct summation (program 1), but

using different function tables (Table III, modes 7 and 8). U_{wave} and the derived force

$$\mathbf{F}_{\mathbf{wave}} = \frac{2q_i}{L^3}\sum_{\mathbf{k}} \frac{\exp\left(-\frac{\pi^2\eta^2\mathbf{k}^2}{L^2}\right)}{\frac{\mathbf{k}^2}{L^2}} \times\left(\sin\frac{2\pi\mathbf{k}\cdot\mathbf{r_i}}{L}\sum_j q_j\cos\frac{2\pi\mathbf{k}\cdot\mathbf{r_j}}{L} - \cos\frac{2\pi\mathbf{k}\cdot\mathbf{r_i}}{L}\sum_j q_j\sin\frac{2\pi\mathbf{k}\cdot\mathbf{r_j}}{L}\right)\frac{\mathbf{k}}{L} \quad (11)$$

are computed by the following scheme:

(0) $a(\mathbf{k}) \leftarrow \dfrac{\exp\left(-\frac{\pi^2\eta^2\mathbf{k}^2}{L^2}\right)}{\frac{\mathbf{k}^2}{L^2}}$ Host

(1) $bs(\mathbf{k}) \leftarrow \sum_j q_j\sin\dfrac{2\pi\mathbf{k}\cdot\mathbf{r_j}}{L}$,

$bc(\mathbf{k}) \leftarrow \sum_j q_j\cos\dfrac{2\pi\mathbf{k}\cdot\mathbf{r_j}}{L}$ GRAPE

(2) $U_{\mathrm{wave}} \leftarrow \dfrac{1}{2\pi L^3}\sum_{\mathbf{k}} a(\mathbf{k})\{bs(\mathbf{k})^2 + bc(\mathbf{k})^2\}$ Host

(3) $\mathbf{cs}(\mathbf{k}) \leftarrow a(\mathbf{k})bs(\mathbf{k})\dfrac{\mathbf{k}}{L}$, $\mathbf{cc}(\mathbf{k}) \leftarrow a(\mathbf{k})bc(\mathbf{k})\dfrac{\mathbf{k}}{L}$ Host

(4) $\mathbf{fs_i} \leftarrow \sum_{\mathbf{k}} \mathbf{cc}(\mathbf{k})\sin\dfrac{2\pi\mathbf{k}\cdot\mathbf{r_i}}{L}$,

$\mathbf{fc_i} \leftarrow \sum_{\mathbf{k}} \mathbf{cs}(\mathbf{k})\cos\dfrac{2\pi\mathbf{k}\cdot\mathbf{r}_i}{L}$ GRAPE

(5) $\mathbf{Fwave_i} \leftarrow \dfrac{2q_i}{L^3}(\mathbf{fs_i} - \mathbf{fc_i})$ Host

Step (0) is performed only at the beginning of the simulation, while (1)–(5) are performed at each time step. The program to compute, for instance, $bs(\mathbf{k})$ of step (2) is as follows.

```
Program for GRAPE (3): EWALD K-SPACE
integer:: Nk          ! number of wave number
                      ! space vectors
real:: k(3, Nk)       ! wave number space vec-
                      ! tors
real:: bs(Nk)         ! bs(k)
integer:: iopt        ! iopt = 1 for sine
                      ! = 2 for cosine

call M1presetmode(9)          ! set Ewald wave-
                              ! space mode
call M1setxjn(1, n, r(1, 1))  ! send coordinates
                              ! of atoms 1−n
                              ! to GRAPE
call M1setqxn(1, n, q(1))     ! send partial
                              ! charge of atoms
                              ! 1−n
iopt = 1                      ! option 1 is for
                              ! computation of
                              ! sine.
call M1execewald(k(1, 1), Nk, n, iopt, bs(1), iovf)
                              ! compute bs(:)
```

M1execewald is similar to M1execn, but it performs the summation defined by eq. (3) rather than eqs. (1) or (2).

MD Software Package PEACH

OVERALL STRUCTURE

The overall structure of PEACH is presented in Figure 2. Like conventional MD software packages, PEACH consists of several program modules: MKDBAS, MKMOL, MKCOR, MKPARA, RUNMD, and ANALMD. MKDBAS constructs a data base of amino acids and nucleotides. MKMOL constructs the topology (connectivity) of the molecule to simulate. MKCOR reads the structural data, adds solvents, and makes the coordinate file for RUNMD and the topology file with solvent information. Then MKPARA assigns the force field parameters and makes the final topology file. RUNMD reads the initial coordinate and topology files thus constructed, and performs MD simulation and energy minimization. ANALMD consists of several programs for analysis of the generated trajectory. Features of PEACH are listed in Table IV.

PEACH is written in FORTRAN90. PEACH was compiled by DEC FORTRAN90 (ver. 3.0) under a Digital Unix (ver. 3.2) environment and was used for most of the computational data presented in this article.

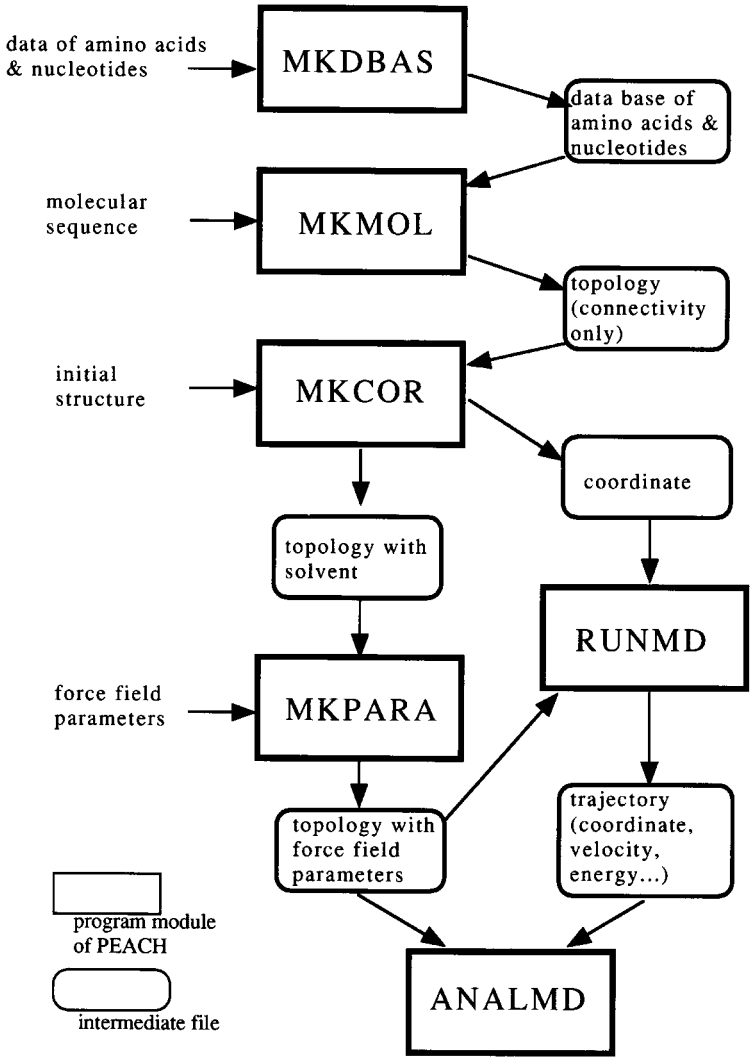

FIGURE 2. Schematic structure of PEACH, which consists of several program modules. The topology and coordinate files were constructed by MKDBAS, MKMOL, MKCOR, and MKPARA, and then used by RUNMD to perform MD simulation. The simulated trajectory was analyzed by ANALMD. The data were transmitted among the modules by intermediate files.

TABLE IV.
Features of PEACH.

Solvent	Cubic box, shell, or sphere
Boundary	Free, periodic, or spherical
Integration	Velocity-Verlet[43] or drRESPA[29]
Ensemble	Microcanonical or canonical (Nose–Hoover[30] or Woodcock[49])
Interactions	Bond, angle, torsion, van der Waals, H bond, electrostatic (Direct or Ewald summation)
Language	FORTRAN 90

INTERACTIONS HANDLED BY PEACH

PEACH currently handles OPLS[33] and several AMBER force field parameter sets.[34–36] Below are the potential functions common to OPLS[33] and Cornell's all atom[36] force fields.

$$U = \sum_{\text{bonds}} K_r(r - r_0)^2 \qquad \text{bond}$$

$$+ \sum_{\text{angles}} K_\theta(\theta - \theta_0)^2 \qquad \text{angle}$$

$$+ \sum_{\text{torsions}} \frac{V_n}{2}[1 + \cos(n\phi - \gamma)] \quad \text{torsion}$$

$$+ \sum_{i>j} \left[\frac{a_{ij}}{r_{ij}^{12}} - \frac{b_{ij}}{r_{ij}^{6}} \right] \quad \text{VDW}$$

$$+ \frac{1}{SC_{14vdW}} \sum_{1-4\ \text{pairs}} \left[\frac{a_{ij}}{r_{ij}^{12}} - \frac{b_{ij}}{r_{ij}^{6}} \right] \quad 1\text{–}4\ \text{VDW}$$

$$+ \sum_{i>j} \frac{q_i q_j}{r_{ij}} \quad \text{Electrostatic}$$

$$+ \frac{1}{SC_{14ELC}} \sum_{1-4\ \text{pairs}} \frac{q_i q_j}{r_{ij}} \quad 1\text{–}4\ \text{Electrostatic} \tag{12}$$

SC_{14vdW} and SC_{14ELC} are the factors to divide vdW and electrostatic interactions between 1 and 4 pairs. The bond, angle, and torsion interactions are called bonded interactions. Their computational costs increase only in proportion to $O(N)$, and they are computed by the host. The electrostatic and vdW interactions are called nonbonded interactions. They are computed by MD-GRAPE because their costs of computation are $O(N^2)$. The Ewald force and potential [eqs. (8)–(11)] are computed for MD in a periodic boundary condition. Also handled is the 10–12 type potential for hydrogen pairs [eq. (5); Table III, H bond), such as was used in the earlier AMBER force field sets.[34,35] Nonbonded interactions are not computed for 1–2 and 1–3 pairs, and those for 1–4 pairs are computed separately by the host. These are called excluded atoms. MD-GRAPE computes interactions between all the atom pairs, and the interactions due to the excluded atoms are subtracted by the host from those by MD-GRAPE.

There is a version of RUNMD that computes nonbonded interactions by a residue based cutoff method without using GRAPE.

MULTIPLE TIME STEP AND THERMOSTAT

In addition to the use of the special-purpose computer, we tried to take advantage of the multiple time step method to enhance the performance. Time scale separation is present in most molecular systems, and application of several time steps, depending on the time scale of the interactions, saves computation time without sacrificing the accuracy. Of several multiple time step methods, we chose Tuckerman's drRESPA (double reversible reference system propagator algorithm)[29] because it is a time-reversible and hence stable algorithm. It has been successfully applied to MD of several molecules including proteins.[37–42] These earlier studies showed that RESPA methods produced the correct dynamics of the molecules faster than the standard MD.

The Nose–Hoover thermostat,[30] which is known to produce a canonical ensemble precisely when applied to systems consisting of a large number of particles, was implemented in PEACH. Below we give a brief derivation of the integrator of the Nose dynamics according to the scheme of Tuckerman et al.[29]

The Nose Hamiltonian H' is expressed as follows for a classical N atom system:

$$H' = \sum_{i=1}^{N} \frac{m_i \mathbf{v_i}^2}{2} + V(\mathbf{r_0} \cdots \mathbf{r_N}) + \frac{p_\eta^2}{2Q} + 3NkT\eta. \tag{13}$$

Here, m_i, $\mathbf{r_i}$, and $\mathbf{v_i}$ are the mass, coordinate, and velocity of atom i. V is the potential energy. T and k are the preset temperature and the Boltzman factor, respectively. Q, η, and p_η are pseudomass, coordinate, and momentum of the Nose particle that governs the temperature. Q is obtained from some characteristic time τ of the simulated system by $Q = 3Nk\tau$. The variables in eq. (13) satisfy the following:

$$\dot{\mathbf{r}}_\mathbf{i} = \mathbf{v_i}, \quad m_i \dot{\mathbf{v}}_\mathbf{i} = -\frac{\partial V}{\partial \mathbf{r_i}} - \frac{p_\eta}{Q} \cdot m_i \mathbf{v_i}, \quad \dot{\eta} = \frac{p_\eta}{Q},$$

$$\dot{p}_\eta = \sum_{i=1}^{N} \frac{m_i \mathbf{v_i}^2}{2} - 3NkT. \tag{14}$$

For simplicity, $\mathbf{v_1} \ldots \mathbf{v_i} \ldots \mathbf{v_N}$ are presented as v, $\mathbf{r_1} \ldots \mathbf{r_i} \ldots \mathbf{r_N}$ as r, and so on.

Although H' is not a classical Hamiltonian in a strict sense, Liouvillian iL for H' can be defined as follows:

$$iL = \nu \frac{\partial}{\partial r} + \frac{f}{m} \frac{\partial}{\partial v} + \frac{p_\eta}{Q} \frac{\partial}{\partial \eta} - \frac{p_\eta}{Q} \nu \frac{\partial}{\partial \nu} + F_\eta \frac{\partial}{\partial p_\eta} \tag{15}$$

Here, $f \equiv -\partial V / \partial r$ and $F\eta \equiv \sum \nu^2 / m - 3NkT$. The Liouvillian (15) is proved to satisfy $iLH' = 0$ by using eq. (14).

By using iL, propagator $U(t)$ is expressed as $U(t) = \exp(iLt)$. When $iL = iL_1 + iL_2$ and $t = n\Delta t$, the propagator is approximated as follows (Trotter

factorization):

$$U(t) \approx \left[\exp\left(iL_2\frac{\Delta t}{2}\right)\exp(iL_1\Delta t)\exp\left(iL_2\frac{\Delta t}{2}\right)\right]^n. \tag{16}$$

There are a number of ways to break down the iL for Nose dynamics [eq. (15)], and therefore we could make many integrator algorithms. To facilitate the implementation to PEACH, we chose a fairly simple algorithm. Let iL_1 be the part of iL irrelevant to the Nose particle:

$$iL_1 = \nu\frac{\partial}{\partial r} + \frac{f}{m}\frac{\partial}{\partial \nu}. \tag{17}$$

Then,

$$iL = iL_1 + \frac{p_\eta}{Q}\frac{\partial}{\partial \eta} - \frac{p_\eta}{Q}\nu\frac{\partial}{\partial \nu} + F_\eta\frac{\partial}{\partial p_\eta}. \tag{18}$$

Using iL (18) and eq. (16) at $n = 1$ repeatedly, the discrete propagator $G(\Delta t)$ for the Nose dynamics is obtained as follows:

$$\begin{aligned} G(\Delta t) = &\exp\left[\frac{\Delta t}{2}F_\eta(\nu)\frac{\partial}{\partial p_\eta}\right]\cdot\exp\left[-\frac{\Delta t}{2}\frac{p_\eta}{Q}\nu\frac{\partial}{\partial \nu}\right] \\ &\cdot\exp\left[\frac{\Delta t}{2}\frac{p_\eta}{Q}\frac{\partial}{\partial \eta}\right]\cdot G_1(\Delta t) \\ &\cdot\exp\left[\frac{\Delta t}{2}\frac{p_\eta}{Q}\frac{\partial}{\partial \eta}\right]\cdot\exp\left[-\frac{\Delta t}{2}\frac{p_\eta}{Q}\nu\frac{\partial}{\partial \nu}\right] \\ &\cdot\exp\left[\frac{\Delta t}{2}F_\eta(\nu)\frac{\partial}{\partial p_\eta}\right], \end{aligned} \tag{19}$$

where $G_1(\Delta t)$ is the propagator due to iL_1 (17):

$$G_1(\Delta t) = \exp(iL_1\Delta t) = \exp\left[\Delta t\left(\nu\frac{\partial}{\partial r} - \frac{f}{m}\frac{\partial}{\partial \nu}\right)\right]. \tag{20}$$

The propagator for the velocity-Verlet algorithm[43] is obtained by factorization as follows:

$$\begin{aligned} G_1(\Delta t) = &\exp\left(\frac{\Delta t}{2}\frac{f}{m}\frac{\partial}{\partial \nu}\right)\cdot\exp\left(\Delta t\nu\frac{\partial}{\partial r}\right) \\ &\cdot\exp\left(\frac{\Delta t}{2}\frac{f}{m}\frac{\partial}{\partial \nu}\right). \end{aligned} \tag{21}$$

When f is divided into f_s (soft), f_m (medium), and f_h (hard) forces,

$$f = f_s + f_m + f_h, \tag{22}$$

the propagator for drRESPA is obtained by using eq. (16):

$$\begin{aligned} G_1(\Delta t) = &\exp\left(\frac{\Delta t}{2}\frac{f_s}{m}\frac{\partial}{\partial \nu}\right)\cdot\left[\exp\left(\frac{\Delta t_m}{2}\frac{f_m}{m}\frac{\partial}{\partial \nu}\right)\right. \\ &\cdot\left\{\exp\left(\frac{\Delta t_h}{2}\frac{f_h}{m}\frac{\partial}{\partial \nu}\right)\cdot\exp\left(\Delta t_h\nu\frac{\partial}{\partial r}\right)\right. \\ &\left.\cdot\exp\left(\frac{\Delta t_h}{2}\frac{f_h}{m}\frac{\partial}{\partial \nu}\right)\right\}^K \\ &\left.\cdot\exp\left(\frac{\Delta t_m}{2}\frac{f_m}{m}\frac{\partial}{\partial \nu}\right)\right]^L\cdot\exp\left(\frac{\Delta t}{2}\frac{f_s}{m}\frac{\partial}{\partial \nu}\right), \end{aligned} \tag{23}$$

where Δt_m and Δt_h are time steps for the medium and hard forces that satisfy $\Delta t = L\Delta t_m$ and $\Delta t_m = K\Delta t_h$.

Integration algorithms are made from the propagators (19) and (21) or (23) by using the following equations:

$$\exp\left(c\frac{\partial}{\partial y}\right)A = A, \quad \exp\left(c\frac{\partial}{\partial y}\right)f(y) = f(y + c),$$
$$\exp\left(cy\frac{\partial}{\partial y}\right)f(y) = f(ye^c). \tag{24}$$

The drRESPA with Nose dynamics obtained by combination of eqs. (19) and (23) is similar to Tuckerman et al.,[29] but somewhat simpler because we separated the integration of the real atoms from that of the Nose particle. Below we give a code excerpt. The time integration part of PEACH is essentially the same as that below:

```
do i = 1, nstep
   p_η = p_η + Δt_s/2 * F_η
   η = η + Δt_s * p_η/Q
   ν = ν * exp(−Δt_s/2 * p_η/Q)
!  ******* begin drRESPA *********
   ν = ν + Δt_s/2 * f_s/m
   do j1 = 1, L
      ν = ν + Δt_m/2 * f_m/m
      do j2 = 1, K
         ν = ν + Δt_η/2 * f_η/m
         r = r + Δt_η * ν
```

```
        call calforce (f_h) ! calculate f_h
        v = v + Δt_h/2 * f_h/m
      end do
      call calforce (f_m)
      ν = ν + Δt_m/2 * f_m/m
    end do
    call calforce (f_s)
   ν = ν + Δt_s/2 * f_s/m
!  ******** end of drRESPA **********
    ν = ν * exp(−Δt_s/2 * p_η/Q)
    call calfehta (F_η) ! calculate F_η
    p_η = p_η + Δt_s/2 * F_η
end do
```

Molecular Dynamics Simulation of HPr

CONDITIONS OF MD

Nanosecond order MD simulations of solvated HPr were performed to demonstrate the performance of the PEACH-GRAPE system. A summary of MD conditions is presented in Table V. We used two typical no cutoff schemes; one is the direct summation within an isolated protein-solvent system, and the other is the Ewald summation in a periodic boundary. We also examined two kinds of ensembles, canonical and microcanonical. Four trajectories were thus generated without nonbonded cutoff by using GRAPE: simNS, simNB, simCS, and simCB. For simN*S* and simC*S*, direct summation of the usual Coulomb interaction was performed in a *S*pherical solvent (Fig. 3a). For simN*B* and simC*B*, Ewald summation was performed in a periodic cubic *B*ox solvent (Fig. 3b). See the next subsection for determination of the condition of the Ewald summation. SimN*S* and simN*B* were performed for 1 ns by *N*ose dynamics at 300 K. simC*S* and simC*B* were started from the 300-ps configurations of simNS and simNB, respectively, and continued for a further 700 ps in *C*lassical dynamics. Another trajectory, simNS_cut, was obtained by the same simulation protocol as simNS, except that a residue-based cutoff scheme (8 Å) was employed for nonbonded interactions. SimNS_cut was performed for 500 ps.

DETERMINATION OF PARAMETERS FOR EWALD METHOD

The precision of the Ewald summation [eqs. (8)–(11)] depends on several parameters: η, k_{max}, and r_{cut}. The parameter η determines the balance between r-space and k-space summations; r-space dominates at large η while k-space dominates at small η. The maximum absolute value of **k**s, k_{max}, determines the precision of k-space summation, and r_{cut} is the cutoff radius for the r-space. The computed values of force and potential should be independent of η at infinite k_{max} and r_{cut}; in practice, however, we have to use as small a k_{max} and r_{cut} as possible to save computation time. We should thus determine η so that the highest precision could be obtained within limited computation time.

We determined the parameters as follows. First, we used a large k_{max} (= 10) for the k-space and considered all the contributions from the 26 image cells surrounding the central cell for the r-space. The potential and force computed at various η by this condition were used as the reference values (U_{ref} and f_{ref}). Then a less time consuming and hence practical condition was applied ($k_{max} = 5$ for the k-space, minimal image for the r-space). The potentials were calculated with various values of η in the practical condition (U_{pra} and f_{pra}), and η was chosen that gave the smallest deviation of U_{pra} and f_{pra} from U_{ref} and f_{ref}.

In general, the potential and force depend on η differently, but we tried to find an optimal η that gave enough precision to both. The relative difference (Δfr) between forces computed in two conditions ($\mathbf{f_1}$ and $\mathbf{f_2}$) was obtained by the following:

$$\Delta f_r = \frac{\left(\frac{1}{N} \sum_{i=1}^{N} |\mathbf{f}_{1,i} - \mathbf{f}_{2,i}|^2 \right)^{1/2}}{\left(\frac{RMS(|\mathbf{F}_1|)^2 + RMS(|\mathbf{f}_2|)^2}{2} \right)^{1/2}} \quad (25)$$

where RMS means root mean square.

First of all, η dependence of the Ewald energy calculated in the reference condition was investigated (Fig. 4, solid line). The values agreed quite well, to within an accuracy of $\sim 10^{-6}$ when $4 \leq \eta \leq 16$. The same was true for the force; the relative difference (Δf_r) between any two η values was smaller than 10^{-6}, except for $\eta = 2$ and 4 (not shown). Thus the potential and force obtained for $6 \leq \eta \leq 16$ were qualified as reference values (U_{ref} and f_{ref}).

Then U_{pra} and f_{pra} were compared with U_{ref} and f_{ref}. U_{pra} agreed with U_{ref} when $6 \leq \eta \leq 8$ with a relative error of $\sim 10^{-4}$ (Fig. 4a). The agreement between f_{pra} and f_{ref} was best at $\eta = 8$, when $\Delta f_r \sim 10^{-4}$ (Fig. 4b). Therefore, we adopted $\eta = 8$ Å for further simulations.

TABLE V.
Details of MD of Hpr.

Initial Structure	1.6 Å Crystal Structure (PDB 1ptf[31])	
Force field	Protein: OPLS[33]	
	Solvent: flexible SPC water[50]	
Integration	0 −10 ps: Velocity-Verlet[43] (Δt = 0.5 fs)	
	10 −10,000 ps: RESPA[29],	
	(Δt_h = 0.5 fs, bonded interactions)	
	(Δt_s = 2.0 fs, nonbonded interactions)	
Size (number of atoms)		
simNS, simNS_cut,	Protein:	801
and simCS	Solvent:	9,333
(Fig. 3a)	Total:	10,134
simNB and simCB	Protein:	801
(Fig. 3b)	Solvent:	9,726
	Total:	10,527
Solvent[a]	(The shortest distance between the protein and the solvent was 2.5 Å.)	
simNS, simNS_cut, and simCS	Sphere centered around the centroid of the protein. Constraint force of 1.5 kcal / Å was imposed so that the radius would be 30.4 Å. The shortest distance between the protein and the surface of the sphere was 10 Å.	
simNB and simCB	Cubic box (48.85 × 48.85 × 48.85 Å); The shortest distance between the protein and the wall was 7 Å	
Ensemble		
simNS, simNS_cut, and simNB	0 −10 ps: Canonical (0 −300 K heating) (Nose −Hoover, τ = 0.1 ps) 10 −1000 ps: Canonical (300 K) (Nose −Hoover, τ = 0.5 ps)	
simCS and simCB[b]	simNS_cut performed for only 500 ps (starting from the 300 ps configuration of simNS or simNB) 300 −1000 ps: microcanonical (classical dynamics)	
Electrostatic interaction		
simNS and simCS	Direct summation without cut-off	
simNS_cut	Residue based cut-off (8 Å) Pair list was updated every 20 fs.	
simNB and simCB	Ewald summation *r*-space: minimal image *k*-space: k_{max} = 5 (515 vectors) η: 8 Å	

DETERMINATION OF TIME STEPS OF DRRESPA

We employed the multiple time step scheme drRESPA to save computation time. Choice of the time steps depends on the force field parameters and the system to simulate, and we tried several combinations of the time steps to find an optimal condition for integration. In evaluation of the stability of the integrator, the relative RMS fluctuation ($\mathrm{RMSF_r}$) of the total energy (E_{tot}) was calculated from the trajectories:

$$\mathrm{RMSF}_r(E_{\mathrm{tot}}) = \left| \frac{\left(\langle (E_{\mathrm{tot}} - \langle E_{\mathrm{tot}} \rangle)^2 \rangle \right)^{1/2}}{\langle E_{\mathrm{tot}} \rangle} \right| \quad (26)$$

Hereafter, $\langle \quad \rangle$ indicates the time average.

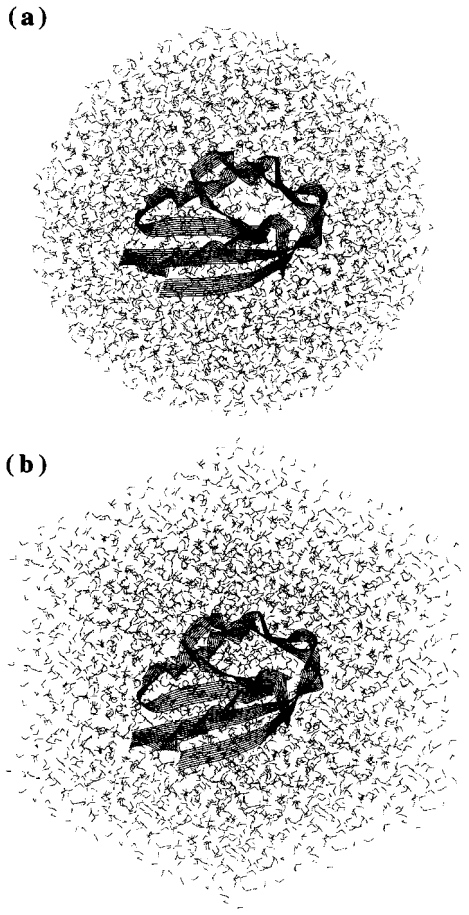

FIGURE 3. Initial structures of MD simulations. The protein, HPr, was placed in (a) spherical water or (b) box water.

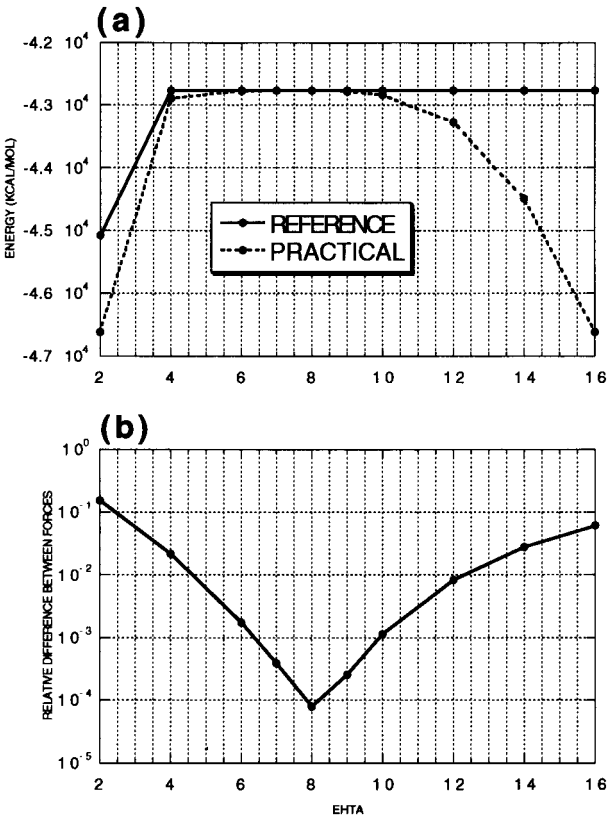

FIGURE 4. Error analysis of the Ewald mode. HPr in a box solvent (Fig. 3b) was equilibrated at 300 K for 1 ps with the minimal image convention for electrostatic, and the final configuration was used for evaluation of the Ewald potential and force. In the reference condition, the interactions within the central cell and those from 26 surrounding cells were considered for r-space, and k_{max} was 10 for *k*-space. In the practical condition, the minimal image convention was used for r-space and k_{max} was 5 for *k*-space. (a) η Dependence of the potential energy computed under the (—) reference condition and the (. . .) practical condition. (b) η Dependence of relative errors [Δf_r, eq. (25)] between forces calculated under the reference and practical conditions.

The interactions in eq. (12) were classified into three as in eq. (22); f_h (bond, angle, and torsion), f_m(vdW and 14 vdW), and f_s (Electrostatic and 14 Electrostatic) forces. The time steps for these forces (Δt_h, Δt_m, Δt_s) were determined as follows. First, the shortest time step Δt_h was fixed to 0.5 fs. This rather large time step for the bonded interactions was chosen because the OPLS force field was used, in which the mass of hydrogen is 3. A shorter time step should be necessary for force fields with a smaller mass for hydrogen. Then MD in the microcanonical ensemble was performed for 5 ps starting from the 10-ps configuration with varying Δt_m and Δt_s (conditions 1–5, Table I). E_{tot} should be conserved in the microcanonical ensemble, but the truncation and round-off errors cause fluctuation in E_{tot}.[44] Hence, $RMSF_r(E_{tot})$ serves as a criterion to test the stability of the integrator. Condition 1 was the reference, in which Δt was 0.5 fs for all the interactions (Table I).

Direct summation of the usual Coulomb interaction was calculated in simulations in the spherical solvent (simNS and simCS). In this case, Δt_s = 4.0 fs gave an unacceptable large energy fluctuation and was inappropriate (Table I, condition 5). Condition 4 gave nearly the same $RMSF_r$ as the reference condition and accelerated the computation speed threefold. Thus condition 4 was used for MD after 10 ps in simNS and later in simCS.

Ewald summation was performed for simulations in the box solvent (simNB and simNC). Conservation of E_{tot} in the reference condition was better than that of the direct summation in the

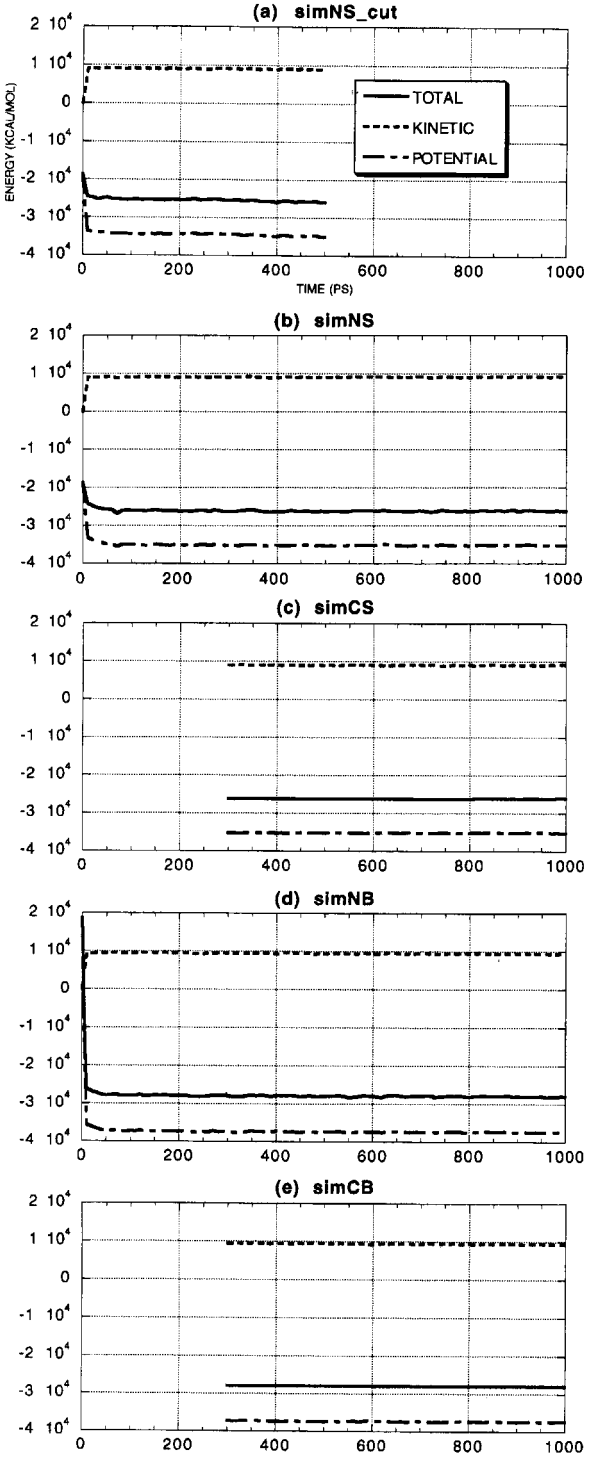

FIGURE 5. Time evolutions of total, kinetic, and potential energies.

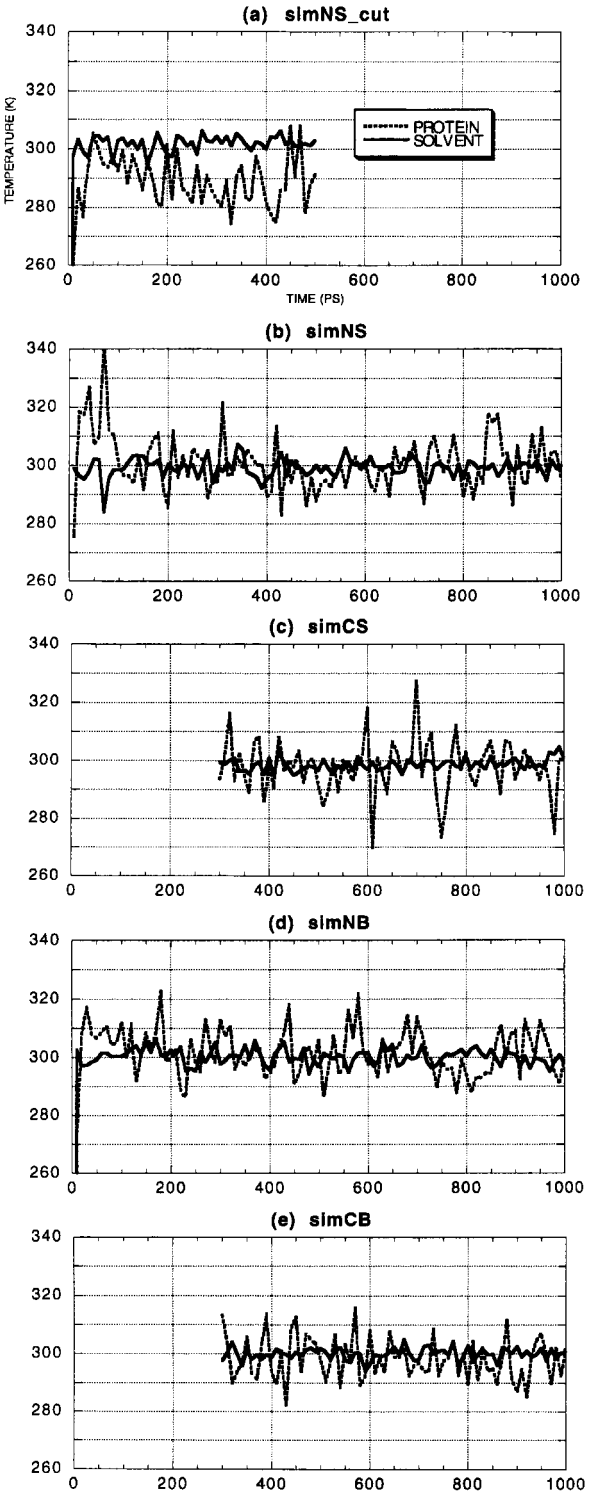

FIGURE 6. Time evolutions of temperatures of the protein and the solvent.

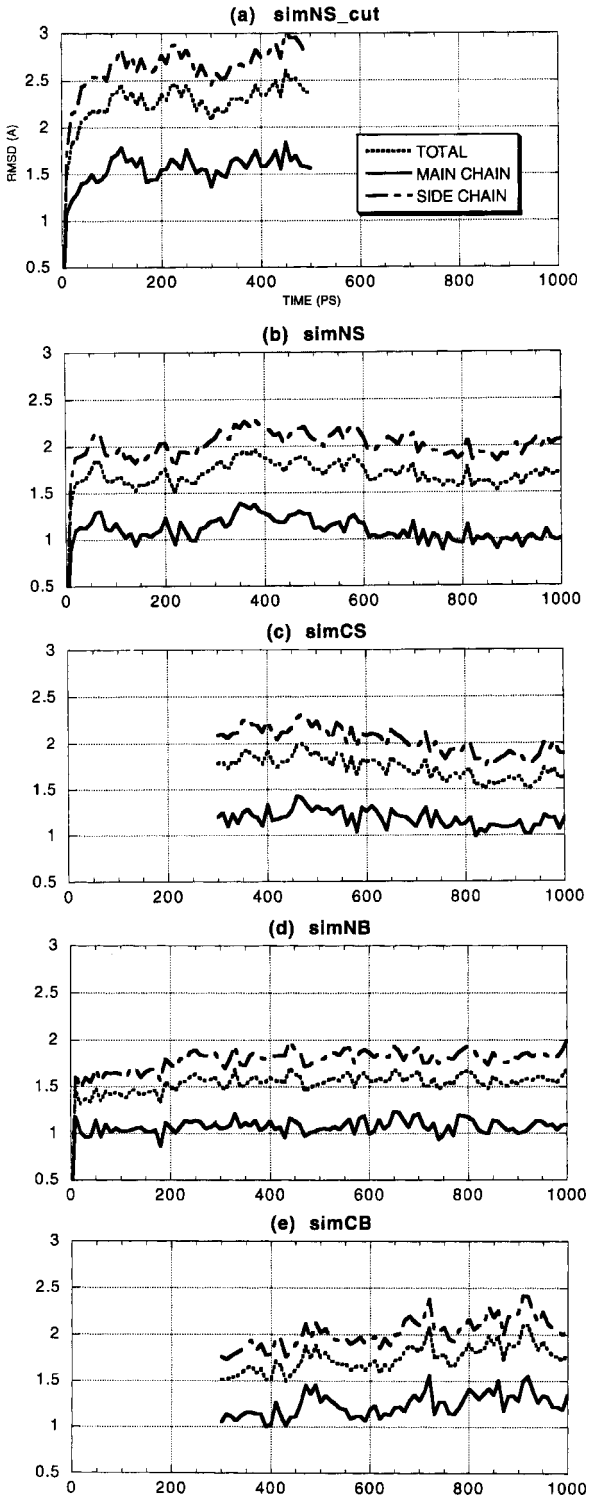

FIGURE 7. Time evolution of RMSD of the simulated structures from the crystal. Hydrogens were omitted from the analyses.

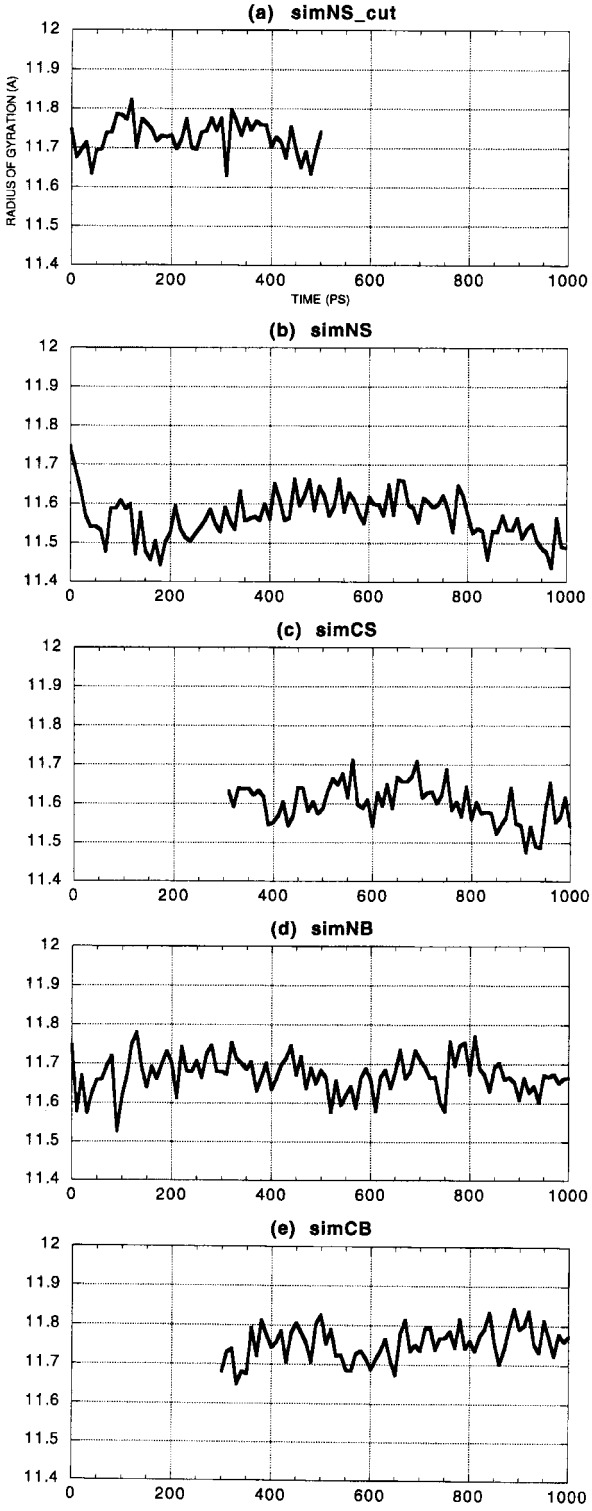

FIGURE 8. Time evolution of radius of gyration [R_g, eq. (27)]. Hydrogens were omitted from the analyses.

spherical water (Table I, condition 1). Condition 4, which gave precision as good as the reference in the spherical solvent, gave poorer precision than the reference in the box solvent. Nevertheless, $\mathrm{RMSF_r}(E_{\mathrm{tot}})$ of 8×10^{-5} was still considered good, compared with 4×10^{-4} obtained under the spherical solvent. Hence, we used condition 4 also in simulations in the periodic box, rather than using more precise conditions (1 or 2) at the expense of the computational time.

Details of the computation time consumed by 1 fs MD at condition 4 were summarized in Table II. Most of the interactions consumed nearly the same computation time in both spherical and box solvent systems because both were about the same size ($\sim$10,000 atoms, Table V). However, an additional *k*-space summation was needed for the latter, which resulted in the longer computation time. The ratios of host:GRAPE computation time were 1:3 and 1:6. Room for improvement of the performance still exists by pararellization of MD-GRAPE.

Computational time for MD simulations with and without truncation of the nonbonded interactions were compared (Table II). See legend d to Table II for details of the cutoff MD at a cutoff radius of 8 Å. The results should not be regarded as strict benchmarks, because the software (PEACH vs. AMBER[45]), integration algorithms (RESPA vs. SHAKE[46]), and compilers (DEC FORTRAN90 vs. FORTRAN77) are different. The difference between compilers, in particular, could be responsible for the difference in the cutoff simulations performed by AMBER and PEACH. Nevertheless, the results show that the PEACH–GRAPE system performs ''no cutoff'' MD at a speed comparable to ordinary cutoff simulations.

ANALYSES OF TRAJECTORIES

The time evolutions of energetic and structural parameters were investigated for the five trajectories using the ANALMD module of PEACH (Figs. 5–8).

The total (E_{tot}), kinetic (E_{kin}), and potential (E_{pot}) energies were plotted (Fig. 5). The energies were quite stable after 100 ps.

The temperature separation, known as ''the hot solvent and cold protein problem,'' is a prominent artifact of the abrupt cutoff of the electrostatic force.[2–5] See ref. 5 for a detailed discussion of the cause of this phenomenon. The temperature separation took place in the trajectory obtained by an abrupt cutoff (simNS_cut; Fig. 6a). The temperature difference of 15 K between the protein and solvent (Fig. 6a) was smaller than previous results of other solvated proteins obtained by an 8-Å cutoff radius. Howard and Kollman[2] reported a difference of 110 K in a simulation of *trp*-repressor. Oda et al.[5] reported a difference of 70 K in a simulation of *Ras* p21. We cannot directly compare our results with theirs, because the protein, the water model, and the force field parameter were different from theirs. However, we suspect that the use of the Nose–Hoover thermostat instead of Berendsen et al.'s[47] could have brought about considerable improvement. Similarly, Guenot and Kollman[3] found that use of a temperature bath by Anderson[48] instead of Berendsen et al. avoids the temperature separation.

In contrast, the temperature separation did not occur in any of the trajectories we generated without the nonbonded cutoff (Fig. 6b–e). The protein had a higher temperature for the first 100 ps of the simNS and simNB (Fig. 6b, d), but after 100 ps both solvent and protein temperatures were kept fairly constant at 300 K for the canonical (simNS and simNB) and microcanonical (simCS and simCB) ensembles. The temperature fluctuation was larger in the protein than in the solvent, obviously because the number of the protein atoms was only about 800 while that of the solvent atoms was more than 10 times larger (Table V).

We also characterized the structural elements of the trajectories by RMS deviation from the crystal structure (RMSD, Fig. 7) and radius of gyration (R_{g}, Fig. 8). R_{g} was calculated as follows:

$$R_{\mathrm{g}} = \left(\frac{I_{\mathrm{g}}}{M} \right)^{1/2} \tag{27}$$

where I_{g} and M stand for the moment of inertia and molecular mass, respectively. Hydrogens were neglected in computation of RMSD and R_{g}.

The trajectory obtained by the nonbonded cutoff (simNS_cut) showed a large RMSD ($>$1.5 Å for the main chain and $>$2.5 Å; Fig. 7a) from the crystal structure. In the no cutoff simulations, however, RMSD fell within 0.9–1.5 Å for the main chain and 1.7–2.5 Å for the side chains (Fig. 7b–d). This is consistent with some previous findings that trajectories obtained by the electrostatic cutoff method usually show larger RMSD from the crystal structure than those obtained without the cutoff.[6,17] All the trajectories, with and without the cutoff, showed similar R_{g} values (11.5–11.8 Å; Fig. 8).

Though the stability of simNS_cut is not unacceptable, we conclude that the trajectories obtained

without the nonbonded cutoff (simNS, simCS, simNB, and simNC) showed better energetic and structural properties than simNS–cut, indicating the superiority of MD without the nonbonded cutoff.

Conclusion

A high performance MD system for biomolecular simulation was constructed by using the software package PEACH and a special-purpose computer GRAPE. This system shows that the electrostatic cutoff, which brings about many problematic artifacts, is no longer required. The PEACH-GRAPE system performs no cutoff MD within reasonable computation time.

MD simulations of a solvated protein were performed by the PEACH-GRAPE system without the nonbonded cutoff using two different ensembles (canonical and microcanonical), two contrasting boundaries (spherical and periodic), and treatment of the electrostatic (direct and Ewald summation). Another trajectory was generated by using the cutoff method for comparison. The trajectories obtained without the cutoff method were considered more stable and reliable than that obtained with the cutoff method.

The PEACH-GRAPE system was thus shown to be useful for accurate simulation of biomolecules. Several MD studies are now in progress in our laboratory, including DNA recognition and ligand–protein interactions.

Acknowledgments

We are grateful to Professor Daiichiro Sugimoto of Tokyo University for supervising the development of MD-GRAPE. We thank Dr. Minoru Saito of Real World Computing Partnership, Dr. Koji Ando of Tsukuba University, Dr. Masuhiro Mikami of the National Institute of Materials and Chemical Research, Dr. Hiroh Miyagawa of Taisho Pharmaceutical Co., Professor Ichiro Yamato of the Science University of Tokyo, and Drs. Izumi Hase and Hiroshi Yokoyama of the Electrotechnical Laboratory for their discussions and support during the development of the software PEACH. The Research Information Processing Station (RIPS) at the Agency of Industrial Science and Technology is acknowledged for computer support. This work was supported by the Research Information Network Grand challenge (RING) Program of the Japanese government.

References

1. M. Karplus and G. A. Petsko, *Nature,* **347**, 631 (1990).
2. A. E. Howard and P. A. Kollman, *Protein Sci.,* **1**, 1173 (1992).
3. J. Guenot and P. A. Kollman, *J. Comput. Chem.,* **14**, 295 (1992).
4. G. E. Arnold and R. L. Ornstein, *Proteins,* **18**, 19 (1994).
5. K. Oda, H. Miyagawa, and K. Kitamura, *Mol. Simul.*, **16**, 167 (1996).
6. M. Saito, *J. Chem. Phys.*, **101**, 4055 (1994).
7. H. Schreiber and O. Steinhauser, *Biochemistry*, **31**, 5856 (1992).
8. M. Saito and R. Tanimura, *Chem. Phys. Lett.,* **236**, 156 (1995).
9. S. G. Kalko, G. Sese, and J. A. Padro, *J. Chem. Phys.*, **104**, 9578 (1996).
10. R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles,* McGraw–Hill, New York, 1981.
11. M. Saito, *Mol. Simul.,* **8**, 321 (1992).
12. J. Shimada, H. Kaneko, and T. Takada, *J. Comput. Chem.,* **14**, 867 (1993).
13. T. A. Darden, D. M. York, and L. G. Pederson, *J. Chem. Phys.*, **98**, 10089 (1993).
14. D. York and W. Yang (1994), *J. Chem. Phys.*, **101**, 3298 (1994).
15. L. Greengard, *Science,* **265**, 909 (1994).
16. P. Procacci, T. Darden, and M. Marchi, *J. Phys. Chem.,* **100**, 10464 (1996).
17. T. Fox and P. A. Kollman, *Proteins,* **25**, 315 (1996).
18. A. F. Bakker, G. H. Gilmer, M. H. Grabow, and K. Thompson, *J. Comput. Phys.,* **90**, 313 (1990).
19. R. Fine, G. Dimmler, and C. Levinthal, *Proteins,* **11**, 242 (1991).
20. T. Ito, T. Fukushige, J. Makino, T. Ebisuzaki, S. K. Okumura, D. Sugimoto, H. Miyagawa, and K. Kitamura, *Proteins,* **20**, 139 (1994).
21. J. Higo, S. Endo, K. Nagayama, T. Ito, T. Fukushige, T. Ebisuzaki, D. Sugimoto, H. Miyagawa, K. Kitamura, and J. Makino, *J. Comput. Chem.,* **15**, 1372 (1994).
22. T. Amisaki, T. Fujiwara, A. Kusumi, H. Miyagawa, and K. Kitamura, *J. Comput. Chem.,* **16**, 1120 (1995).
23. J. Makino and Y. Funato, *Publ. Astron. Soc. Jpn.,* **45**, 279 (1993).
24. M. Taiji, T. Fukushige, J. Makino, T. Ebisuzaki, and D. Sugimoto, In *Proceedings of the 6th Joint EPS-APS International Conference on Physics Computing*, R. Gruber and M. Tomassini, Eds., European Physical Society, Geneva, 1994, p. 67.
25. D. Sugimoto, Y. Chikada, J. Makino, T. Ito, T. Ebisuzaki, and M. Umemura, *Nature,* **345**, 33 (1990).
26. T. Ebisuzaki, J. Makino, T. Fukushige, M. Taiji, D. Sugimoto, T. Ito, and S. K. Okumura, *Publ. Astron. Soc. Jpn.,* **45**, 269 (1993).

27. Y. Komeiji, H. Yokoyama, M. Uebayasi, M. Taiji, T. Fukushige, D. Sugimoto, R. Takata, A. Shimizu, and K. Itsukashi, In *Pacific Symposium on Biocomputing 96*, L. Hunter and T. E. Klein, Eds., World Scientific, Singapore, 1995, p. 472.
28. P. Ewald, *Ann. Phys.,* **64**, 253 (1921).
29. M. Tuckerman, B. J. Berne, and G. J. Martyna, *J. Chem. Phys.,* **97**, 1990 (1992).
30. S. Nose, *Prog. Theor. Phys. Suppl.,* **103**, 1 (1991).
31. Z. Jia, M. Vandonselaar, J. W. Quail, and L. T. J. Delbaere, *Nature,* **36**, 94 (1993).
32. T. Fukushige, J. Makino, T. Ito, S. K. Okumura, T. Ebisuzaki, and D. Sugimoto, *Publ. Astron. Soc. Jpn.,* **45**, 361 (1993).
33. W. L. Jorgensen and J. Tirado−Rives, *J. Am. Chem. Soc.,* **110**, 1657 (1988).
34. S. J. Weiner, P. A. Kollman, D. A. Case, U. C. Singh, C. Ghio, G. Alagona, and P. Weiner, *J. Am. Chem. Soc.,* **106**, 765 (1984).
35. S. J. Weiner, P. A. Kollman, D. T. Nguyen, and D. A. Case, *J. Comput. Chem.,* **7**, 230 (1986).
36. W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz, D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, and P. A. Kollman, *J. Am. Chem. Soc.,* **117**, 5179 (1995).
37. M. E. Tuckerman and W. Langel, *J. Chem. Phys.,* **100**, 6368 (1994).
38. P. Procacci and B. J. Berne, *J. Chem. Phys.,* **101**, 2421 (1994).
39. D. D. Humphreys, R. A. Friesner, and B. J. Berne, *J. Phys. Chem.,* **98**, 6885 (1994).
40. M. Watanabe and M. Karplus, *J. Phys. Chem.,* **101**, 2421 (1994).
41. M. Watanabe and M. Karplus, *J. Phys. Chem.,* **99**, 5680 (1995).
42. S. T. Cui, P. T. Cummings, and H. D. Cochran, *J. Chem. Phys.,* **104**, 255 (1996).
43. W. C. Swope, H. C. Andersen, P. H. Behrens, and K. R. Wilson, *J. Chem. Phys.,* **76**, 637 (1982).
44. J. M. Haile, *Molecular Dynamics Simulation—Elementary Methods*, Wiley, New York, 1992.
45. D. A. Pearlman, D. A. Case, J. W. Caldwell, W. S. Ross, T. E. Cheatham III, D. M. Ferguson, G. L. Seibel, U. C. Singh, P. K. Weiner, and P. A. Kollman, *AMBER 4.1*, University of California, San Francisco, 1995.
46. J. Ryckaert, G. Ciccotti, and H. J. C. Berendsen, *J. Comput. Chem.,* **23**, 327 (1977).
47. H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. DiNola, and J. R. Haak, *J. Chem. Phys.*, **81**, 36 (1984).
48. H. C. Anderson, *J. Chem. Phys.*, **72**, 2384 (1980).
49. L. V. Woodcock, *Chem. Phys. Lett.,* **10**, 257 (1971).
50. L. X. Dang, and B. M. Pettitt, *J. Phys. Chem.,* **91**, 3349 (1987).